

Start of Lecture 4: DOMAINS: Scripts – Human Behaviour

3.4.1. Routes as Scripts

3.4.1.1. Paths

92. A path is a triple:

- (a) a hub identifier, h_i , a link identifier, l_j , and another hub identifier, h_k , distinct from h_i ,
- (b) such that there is a link ℓ with identifier l_j in a net n such that $\{h_i, h_k\}$ are the hub identifiers that can be observed from ℓ .

type

92. $\text{Pth} = \text{HI} \times \text{LI} \times \text{HI}$

axiom

92(a). $\forall (hi, li, hi): \text{Pth} \cdot \exists n: \mathbb{N}, l: \text{L} \cdot l \in \text{obs_Ls}(n) \Rightarrow$

92(b). $\text{obs_LI}(l) = li \wedge \text{obs_HIs}(l) = \{hi, hi\}$

3.4. Scripts

Definition: Scripts.

- A script is plan of action.
- By a domain script we shall, more specifically, understand
 - the structured, almost, if not outright,
 - formally expressed, wording of a set of
 - rules and regulations.
- See also
 - *license* and
 - *contract*.

Definitions follow.

93. From a net one can extract all its paths:

- (a) if l is a link of the net,
- (b) l_j its identifier,
- (c) $\{h_i, h_k\}$ the identifiers of its connected hubs,
- (d) then (h_i, l_j, h_k) and (h_k, l_j, h_j) are paths of the net.

value

93. $\text{paths}: \mathbb{N} \rightarrow \text{Pth-set}$

93(a). $\text{paths}(n) \equiv$

93(d). $\{(hi, lj, hk), (hk, lj, hi) \mid l: \text{L}, lj: \text{LI}, hi, hk: \text{HI} \cdot l \in \text{obs_Ls}(n) \wedge$

93(b). $lj = \text{obs_LI}(l) \wedge$

93(c). $\{hi, hk\} = \text{obs_HIs}(l)\}$

94. From a net descriptor one can (likewise) extract all its paths:

- (a) Let h_i, h_k be any two distinct hub identifiers of the net descriptor (definition set),
- (b) such that they both map into a link identifier l_j ,
- (c) then (h_i, l_j, h_k) and (h_k, l_j, h_i) are paths of the net.

value

93. paths: $ND \rightarrow Pth\text{-set}$

93. paths(nd) \equiv

94(a). $\{(hi, lj, hk), (hk, lj, hi) \mid hi, hk:HI, lj:LI \cdot hi \neq hk \wedge \{hi, hk\} \subseteq \mathbf{dom} \text{ nd} \Rightarrow$

94(b). $lj \in \mathbf{dom} \text{ nd}(hi) \cap \mathbf{dom} \text{ nd}(hk)\}$

96. From a net, n , we can generate the possibly infinite set of finite and possibly infinite routes:

- (a) $\langle \rangle$ is a path (**basis clause 1**);
- (b) if p is a path of n then $\langle p \rangle$ is a path of n (**basis clause 2**);
- (c) if r and r' are non-empty routes of n
 - i. and the last h_i of r is the same as the first h_j of r'
 - ii. then the concatenation of r and r' is a route (**induction clause**).
- (d) Only such routes which can be formed by a (finite, respectively infinite) application of basis clauses Items 96(a) and 96(b) and induction clause Item 96(c) are routes (**extremal clause**).

3.4.1.2. Routes

95. A route of a net is a sequence of zero, one or more paths such that

- (a) all paths of a route are paths of the net and
- (b) adjacent paths in the sequence “share” hub identifiers.

type

95. $R = Pth^*$

axiom

95. $\forall r:R, \exists n:N \cdot$

95(a). $\mathbf{elems} \ r \subseteq \mathbf{paths}(n) \wedge$

95(b). $\forall i:\mathbf{Nat} \cdot \{i, i+1\} \subseteq \mathbf{inds} \ r \Rightarrow$

95(b). $\mathbf{let} \ (_, _, hi)=r(i), (hi', _, _)=r(i+1) \ \mathbf{in} \ hi=hi' \ \mathbf{end}$

value

96. routes: $N|ND \rightarrow R\text{-infset}$

96. routes(nond) \equiv

96(a). $\mathbf{let} \ rs = \{\langle \rangle\} \cup$

96(b). $\{\langle p \rangle \mid p:Pth \cdot p \in \mathbf{paths}(nond)\} \cup$

96((c))ii. $\{r \hat{\ } r' \mid r, r':R \cdot r \in rs \wedge r' \in rs \wedge$

96((c))i. $\exists hi, hi', hi'', hi''':H, li:LI \cdot$

96((c))i. $r=r'' \hat{\ } \langle (hi, li, hi') \rangle \wedge r'=\langle (hi'', li', hi''') \rangle \hat{\ } r''' \wedge$

96((c))i. $hi'=hi'' \} \ \mathbf{in}$

96(d). $rs \ \mathbf{end}$

3.4.2. Bus Timetables as Scripts

3.4.2.1. Buses

97. Buses are vehicles,

98. with bus identifiers being the same as vehicle identifiers.

type

97. B

98. $BI \subseteq VI$

3.4.2.2. Bus Stops

99. A link bus stop indicates the link (by its identifier), the from and to hub identifiers, and the fraction “down the link” from the from to the to hub identifiers.

type

99. $BS = \text{mkL_BS}(\text{sel_fhi}:HI, \text{sel_li}:LI, \text{sel_f}:F, \text{sel_thi}:HI)$

value

$n:N$

type

100. $BSL = BS^*$

101. $BR = \{(r, bsl) : (R \times BSL) \cdot \text{wf_BR}(r, bsl)\}$

value

101. $\text{wf_BR}: BR \rightarrow \mathbf{Bool}$

101. $\text{wf_BR}(r, bsl) \equiv \exists n:N, r:R \cdot r \in \text{routes}(n) \wedge \text{is_embedded_in}(r, bsl)$

101(a). $\text{is_embedded_in}: BR \rightarrow \mathbf{Bool}$

101(a). $\text{is_embedded_in}(r, bsl) \equiv$

101(b). $\exists il:\mathbf{Nat}^* \cdot \text{len } il = \text{len } bsl \wedge \text{inds } il \subseteq \text{inds } r \wedge \text{ascending}(il) \Rightarrow$

101(c). $\forall i:\mathbf{Nat} \cdot i \in \text{inds } il \Rightarrow$

101(c). **let** $(hi, lj, hk) = r(il(i)), (hi', lj', f, hk') = bsl(i)$ **in**

101(c). $hi = hi' \wedge lj = lj' \wedge hk = hk'$ **end** \wedge

102. $\forall i:\mathbf{Nat} \cdot \{i, i+1\} \subseteq \text{inds } il \Rightarrow$

102. **let** $(hi, lj, f, hk) = bsl(i), (hi', lj', f, hk') = bsl(i+1)$ **in**

102. $hi = hi' \wedge lj = lj' \wedge hk = hk' \Rightarrow f < f'$ **end**

ascending: $\mathbf{Nat}^* \rightarrow \mathbf{Bool}$, $\text{ascending}(il) \equiv \forall i:\mathbf{Nat} \cdot \{i, i+1\} \subseteq \text{inds } il \Rightarrow il(i) \leq il(i+1)$

3.4.2.3. Bus Routes

100. A bus stop list is a sequence of two or more bus stops, bsl .

101. A bus route, br , is a pair of a net route, r , and a bus stop list, bsl , such that route r is a route of n and such that bsl is embedded in r . If

(a) there exists an index list, il , of ascending indices of the route r and of the length of bsl

(b) such that the i th path of r

(c) share from and to hub identifiers and link identifier with the $il(i)$ th bus stop of bsl

then bsl is embedded in r .

102. We must allow for two or more stops along a bus route to be adjacent on the same link — in which case the corresponding fractions must likewise be ascending.

3.4.2.4. Bus Schedule

103. A timed bus stop is a pair of a time and a bus stop.

104. A timed bus stop list is a sequence of timed bus stops.

105. A bus schedule is a pair of a route and a timed bus stop list such that

- there is a net of which the routes is indeed a route,
- the bus stop list of the timed bus stop list is embedded in the route, and
- ‘later’ listed bus stops register later times.

106. `SimpleBusSchedules` remove routes from `BusRoutes`.

type

103. $TBS :: sel_T:T \ sel_bs:BS$

104. $TBSL = TBS^*$

105. $BusSched = \{|(r,tbsl):(R \times TBSL) \cdot wf_BusSched(r,tbsl)|\}$

value

105. $wf_BusSched: BusSched \rightarrow \mathbf{Bool}$

105. $wf_BusSched(r,tbsl) \equiv$

105. $\exists n:N \cdot r \in routes(n)$

105. $\wedge \mathbf{let} \ bsl:SBS = \langle sel_BS(tbsl(i)) | i:[1..len \ tbsl] \rangle \mathbf{in} \ is_embedded_in(r,bsl) \mathbf{end}$

105. $\wedge \forall i:\mathbf{Nat} \cdot \{i,i+1\} \subseteq \mathbf{inds} \ tbsl \Rightarrow sel_T(tbsl(i)) < sel_T(tbsl(i+1))$

type

106. $SBS = \{|bsl:BS^* \cdot \exists n:N, r:R \cdot r \in routes(n) \wedge is_embedded_in(r,bsl)|\}$

type

107. $BLNm$

value

108. $same_bus_schedule: BusSched \times BusSched \rightarrow \mathbf{Bool}$

108. $same_bus_schedule((r1,bt11),(r2,bt12)) \equiv$

108. $r1 = r2 \wedge \mathbf{len} \ bt11 = \mathbf{len} \ bt12 \wedge$

108. $\langle sel_BS(bt11(i)) | i:[1..len \ bt11] \rangle = \langle sel_BS(bt12(i)) | i:[1..len \ bt12] \rangle$

type

109. BNo

110. $RBS :: sel_R:R \ sel_btbl:(BNo \rightsquigarrow SBS)$

111. $TBL = BLNm \rightsquigarrow RBS$

112. $TT = ND \times TBL$

113. $TT = \{|tt:TT \cdot wf_TT(tt)|\}$

3.4.2.5. Timetable

The concept of a bus line captures all those bus schedules which ply the same bus route but at different times. A timetable is made up from distinctly named bus lines.

107. A bus line has a unique bus line name.

108. We say that two bus schedules are the same if they are based on the same route and if they differ only in their times.

109. Each of the different bus routes of a bus line has a unique bus number.

110. A route bus schedule pairs a route with simple bus schedules for each of a number of busses (identified by their bus number).

111. A bus timetable (listing, map) maps bus line names to route bus schedules.

112. A timetable is a pair, a net and a table.

113. A well-formed timetable must satisfy same bus schedules within each bus line

114. All bus numbers are distinct across bus lines.

value

113. $wf_TT: TT' \rightarrow \mathbf{Bool}$

113. $wf_TT(_,tbl) \equiv$

113. $\forall bln:BLNm \cdot bln \in \mathbf{dom} \ tbl \Rightarrow$

113. $\forall bno,bno':BNo \cdot \{bno,bno'\} \subseteq \mathbf{dom} \ sel_btbl(tbl(blno)) \Rightarrow$

113. $same_bus_schedule(sel_R(tbl(blno)),sel_btbl(tbl(blno))(bno),$

113. $sel_R(tbl(blno)),sel_btbl(tbl(blno))(bno')) \wedge$

114. $\forall bln',bln'':BLNm \cdot \{bln',bln''\} \subseteq \mathbf{dom} \ tbl \wedge bln' \neq bln'' \Rightarrow$

114. $\mathbf{dom} \ sel_btbl(tbl(bln')) \cap \mathbf{dom} \ sel_btbl(tbl(bln'')) = \{\}$

3.4.3. Route and Bus Timetable Denotations

- What are routes and bus timetables scripting ?
- Routes (list of connected link traversal designations) script that one may transport people or freight along the sequence of designated links.
- Bus timetables script (at least) two things:
 - the set of bus traffics on the net which satisfy the bus timetable, and
 - information that potential and actual bus passengers may, within some measure of statistics (and probability), rely upon for their bus transport.

3.4.4. Licenses and Contracts

Definition: License.

- *A license is*
 - a *script*
 - specifically expressing a permission to act;
 - is freedom of action;
 - is a permission granted by competent authority to engage in a business or occupation or in an activity otherwise unlawful;
 - a document, plate, or tag evidencing a license granted;
 - a grant by the holder of a copyright or patent to another of any of the rights embodied in the copyright or patent short of an assignment of all rights.

Licenses appear more to have morally than legally binding poser.

- Here, we shall not develop the idea of bus timetables denoting certain traffics.
 - Instead we refer to our previously sketched model of traffics (Sect. , Pages 89–97).
- Route (designations) and bus timetables
 - script potential and actual route travels, respectively
 - script the dispatch of buses and their travelling.
- Bus timetables can also be seen as a form of contracts
 - between the bus operators offering the bus services
 - and potential and actual passengers,
 - with the contract promising timely transport.
- In the next section, Sect. , we shall sketch a language of bus service contracts and bus service actions implied by such contracts.

Definition: Contract.

- *A contract*
 - is a special kind of license
 - specifically expressing a legally binding agreement between two or more parties —
 - hence a document describing the conditions of the contract;
 - a contract is business arrangement for the supply of goods or services at fixed prices, times and locations.

- In software development a contract specifies what is to be developed:
 - (1) a *domain description*,
 - (2) a *requirements prescription*, or
 - (3) a *software design*;
 or a combination of these (1–2, 2–3, 1–3).

- For a comprehensive treatment of licenses and contracts we refer to [Chapter 10, Sect. 10.6 (Pages 309–326) [jaist-db10]][jaist-mono].
- We shall illustrate fragments of a language for bus service contracts.
- The background for the bus contract language is the following.
 - In many large cities around Europe the city or provincial government secures public transport in the form of bus services operated by many different private companies.
 - Earlier lectures illustrated the concept of bus (service) timetables.
 - The bus services implied by such a timetable, for a city area — with surrounding suburbs etc. — need not be implemented by just one company, but can be contracted, by the city government public transport office, to several companies, each taking care of a subset of the timetable.

- A contract further specifies
 - how it might, or must be developed;
 - criteria for acceptance of what has been developed;
 - delivery dates for the developed items;
 - who the “parties” to the contract are:
 - * the *client* and
 - * the *developer*, etc.

- Different bus operators then take care of non-overlapping parts and all take care of the full timetable.
- It may even be that extra buses need be scheduled, on the fly, in connection with major sports or concert or other events.
- Bus operators may experience vehicle breakdowns or bus driver shortages and may be forced to subcontract other, even otherwise competing bus operators to “step in” and alleviate the problem.

3.4.4.1. Contracts

Schematically we may represent a bus contract as follows:

Contract cn **between contractee** ci **and contractor** cj :

This contract contracts cj **in the period** $[t, t']$ **to**

perform the following services with respect to timetable tt :

operate bus lines $\{bl_{j_1}, bl_{j_2}, \dots, bl_{j_n}\}$

subject to the following occasional exceptions:

cancellation of bus tours:

$\{(bl_{j_a}, \{bno_{a_1}, \dots, bno_{a_m}\}), \dots\}$ **subject to conditions** cbt

insertion of bus tours on lines

$\{bl_{j_\alpha}, bl_{j_\beta}, \dots, bl_{j_\gamma}\}$ **subject to conditions** ibt

subcontracting bus tours on lines

$\{bl_{j_\delta}, bl_{j_\phi}, \dots, bl_{j_\omega}\}$ **subject to conditions** $scbt$.

We abstract the above quoted “one or more of three kinds of exceptions” as one possibly empty clause for each of these alternatives.

119. A bus contract now contains a header, a timetable, the subject bus lines and the exceptions,

120. such that

(a) line names mentioned in the contract are those of the bus lines of the timetable, and

(b) bus (tour) numbers are those of the appropriate bus lines in the timetable.

121. The calendar period is for at least one full day, midnight to midnight.

122. A named contract is a pair of a contract name and a contract.

115. A bus contract has a header with the distinct names of a contractee and a contractor and a time interval.

116. A bus contract presents a timetable.

117. A bus contract presents a set of bus lines (by their identifiers) such that these are in the timetable.

118. And a bus contract may list one or more of three kinds of “exceptions”:

(a) cancellation of one or more named bus tours on one or more bus lines subject to certain (specified) conditions;

(b) insertion of one or more extra bus tours on one or more bus lines subject to certain (specified) conditions;

(c) subcontracting one or more unspecified bus tours on one or more bus lines subject to certain (specified) conditions — to further unspecified contractors.

type

115. CN_m, CId, D, T, CON

115. $CH = CId \times CId \times (D \times D)$

116. $CT = TT$

117. $CLs = BLN_m\text{-set}$

118. $CE = (CA \times IN \times SC) \times CON$

118(a). $CA = BLN_m \xrightarrow{m} BNo\text{-set}$

118(b). $IN = BLN_m \xrightarrow{m} BNo\text{-set}$

118(c). $SC = BLN_m\text{-set}$

119. $CO' = CH \times CT \times CLs \times CE$

120. $CO = \{|co:CO'\text{-wf}.CO(co)|\}$

122. $NCO = CN_m \times CO$

value

120. $\text{wf_CO}: \text{CO}' \rightarrow \mathbf{Bool}$
120. $\text{wf_CO}((\text{ce}, \text{cr}, (\text{d}, \text{d}')), (\text{nd}, \text{tbl}), \text{cls}, ((\text{blns}, \text{blns}', \text{bls}), \text{con})) \equiv$
117. $\text{ce} \neq \text{cr} \wedge$
- 120(a). $\text{cls} \subseteq \mathbf{dom} \text{tbl} \wedge$
- 120(b). $\forall \text{bli}, \text{bli}': \text{BLNm} \cdot \text{bli} \in \mathbf{dom} \text{blns} \wedge \text{bli}' \in \mathbf{dom} \text{blns}' \Rightarrow$
- 120(a). $\{\text{bli}, \text{bli}'\} \subseteq \mathbf{dom} \text{tbl} \wedge$
- 120(b). $\text{blns}(\text{bli}) \cup \text{blns}'(\text{bli}') \subseteq \mathbf{dom} \text{sel_btbtbl}(\text{tbl}(\text{bli})) \wedge$
- 120(a). $\text{bls} \subset \mathbf{dom} \text{tbl} \wedge$
121. $\text{d} < \text{d}'$

123. A bus operator action is either a **commence**, a **cancellation**, an **insertion** or a **subcontracting** action. All actions refer to the (name of) the **contract** with respect to which the action is contracted.
- (a) A **commence** action designator states the bus line concerned and the bus number of that line.
- (b) A **cancellation** action designator states the bus line concerned and the bus number of that line.
- (c) An **insertion** action designator states the bus line concerned and the bus number of that line — for which an extra bus is to be inserted.⁴
- (d) A **subcontracting** action designator, besides the name of the contract with respect to which the subcontract is a subcontract, state a named contract (whose contract name is unique).

⁴The insertion of buses in connection with either unscheduled or extraordinary (sports, concerts, etc.) events can be handled by special, initial contracts.

3.4.4.2. Contractual Actions

For contract cn commence bus tour, line: bli and bus no.: bno

For contract cn cancel bus tour, line: bli and bus no.: bno

For contract cn insert extra bus tour, line: bli and bus no.: bno

Subcontract with respect to contract cn the following:

Contract cn' : for the calendar period $[\text{d}, \text{d}']$ contractee ci contracts contractor cj to perform the following services with respect to timetable tt :

operate bus lines $\{\text{blj}_1, \text{blj}_2, \dots, \text{blj}_n\}$

subject to the following occasional exceptions:

cancellation of bus tours:

$\{(\text{blj}_i, \{\text{bno}_{c_1}, \dots, \text{bno}_{c_m}\}), \dots\}$ subject to conditions cbt

insertion of bus tours on lines

$\{(\text{blj}_i, \{\text{bno}_{i_1}, \dots, \text{bno}_{i_n}\}), \dots\}$ subject to conditions ibt

subcontracting bus tours on lines

$\{\text{blj}_\delta, \text{blj}_\phi, \dots, \text{blj}_\omega\}$ subject to conditions scht .

type

123. $\text{Act} = \text{Com} \mid \text{Can} \mid \text{Ins} \mid \text{Sub}$

123(a). $\text{Com} == \text{mkCom}(\text{sel_cn}: \text{CNm}, \text{sel_bli}: \text{BLNm}, \text{sel_bno}: \text{BNo})$

123(b). $\text{Can} == \text{mkCan}(\text{sel_cn}: \text{CNm}, \text{sel_bli}: \text{BLNm}, \text{sel_bno}: \text{BNo})$

123(c). $\text{Ins} == \text{mkIns}(\text{sel_cn}: \text{CNm}, \text{sel_bli}: \text{BLNm}, \text{sel_bno}: \text{BNo})$

123(d). $\text{Sub} == \text{mkSub}(\text{sel_cn}: \text{CNm}, \text{sel_con}: \text{NCO})$

3.4.4.3. Wellformedness of Contractual Actions

124. In order to express wellformedness conditions, that is, pre-conditions, for the action designators we introduce a **context** which map contract names to contracts.

125. Wellformedness of a contract is now expressed with respect to a context.

type

124. $CTX = CNm \rightsquigarrow CO$

value

125. $wf_Act: Act \rightarrow CTX \rightarrow \mathbf{Bool}$

127. **cancellation** and **insertion** commands have the same static wellformedness conditions as have **commence** command.

127. $wf_Act(mkCan(cnm,bln,bno))(ctx) \equiv wf_Act(mkCom(cnm,bln,bno))(ctx)$

127. $wf_Act(mkIns(cnm,bln,bno))(ctx) \equiv wf_Act(mkCom(cnm,bln,bno))(ctx)$

- Let a defined **cnm** entry in **ctx** be a contract:
 $((ce,cr),(nd,tbl),cls,(blns,bls,bls'),(d,d'))$.

126. If **cmd** is a **commence** command $mkCom(cnm,bln,bno)$, then

- contract name **cnm** must be defined in context **ctx**;
- bus line name **bln** must be defined in the contract, that is, in **cls**, and
- bus number **bno** must be defined in the bus table part of table **tbl**.

126. $wf_Act(mkCom(cnm,bln,bno))(ctx) \equiv$

126(a). $cnm \in \mathbf{dom} \ ctx \wedge$

126. **let** $((ce,cr),(nd,tbl),cls,(blns,bls,bls'),(d,d')) = ctx(cnm)$ **in**

126(b). $bln \in cls \wedge$

126(c). $bno \in \mathbf{dom} \ sel_btbl(tbl(bl))$ **end**

128. If **cmd** is a **subcontract** command then

Let the subcontract command and the **cnm** named contract in **ctx** be

$mkSub(cnm,nco:(cnm',(ce',cr',(d'',d''')), (nd',tbl'),cls',(blns',bls'',bls'''))$

respectively $((ce,cr,(d,d')), (nd,tbl), cls, (blns,bls,bls'))$.

- contract name **cnm** must be defined in context **ctx**;
- contract name **cnm'** must not be defined in context **ctx**;
- the calendar period of the subcontract must be within that of the contract from which it derives;
- the net descriptors **nd** and **nd'** must be identical;
- the tables **tbl** and **tbl'** and must be identical and
- the set, **cls'**, of bus line names that are the scope of the subcontracting must be a subset of **bls'**.

128. $\text{wf_Act}(\text{mkSub}(\text{cnm}, \text{nco}:(\text{cnm}', \text{co}:(\text{ce}', \text{cr}', (\text{d}'', \text{d}'''), (\text{nd}', \text{tbl}'), \text{cls}', (\text{blns}', \text{blns}'', \text{bls}''')))))))(\text{ctx})$
 128(a). $\text{cnm} \in \mathbf{dom} \text{ ctx} \wedge$
 128. **let** $\text{co}' = ((\text{ce}, \text{cr}, (\text{d}, \text{d}')), (\text{nd}, \text{tbl}), \text{cls}, (\text{blns}, \text{blns}', \text{bls}')) = \text{ctx}(\text{cnm})$ **in**
 128(b). $\text{cnm}' \notin \mathbf{dom} \text{ tbl} \wedge$
 128(c). $\text{d} \leq \text{d}'' \leq \text{d}''' \leq \text{d}' \wedge$
 128(d). $\text{nd}' = \text{nd} \wedge$
 128(e). $\text{tbl}' = \text{tbl} \wedge$
 128(f). $\text{cls}' \subseteq \text{bls}'$ **end**

- Wellformedness of contracts, $\text{wf_CO}(\text{co})$ and $\text{wf_CO}(\text{co}')$, secures other constraints.

3.5. Management and Organisation

Definition: Management.

- *Management is about resources:*
 - *their acquisition,*
 - *scheduling (over time),*
 - *allocation (over locations),*
 - *deployment (in performing actions) and*
 - *disposal (“retirement”).*
- *We distinguish between*
 - *board-directed,*
 - *strategic,*
 - *tactical and*
 - *operational**actions.*

- We do not here bring any narrated or formalised description of the semantics of contracts and actions.
- First such a description would be rather lengthy.
- Secondly a specification would be more of a requirements prescription.

- *Board-directed actions target mainly financial resources: obtaining new funds through conversion of goodwill into financial resources, acquiring and selling “competing” or “supplementary” business units.*
- *Strategic actions convert financial resources into production, service supplies and resources and vice-versa — and in this these actions schedule availability of such resources.*
- *Tactical actions mainly allocate resources.*
- *Operational actions order, monitor and control the deployment of resources in the performance of actions.*

Definition: Organisation.

- *Organisation is about*
 - the “grand scale”,
 - * executive and strategic
 - * national, continental or global (world wide)
 - (i) *allocation* of major resource (e.g., business) units, whether in a hierarchical, in a matrix, or in some other organigram-specified structure,
 - (ii) *as well as the clearly defined relations* (which information, decisions and actions are transferred) between these units, and
 - (iii) *organisational dynamics*.

3.5.1. Transport System Examples

We shall only present sketchy examples of management and organisation.

- **Executive actions:**
 - Deciding on major re-organisation of a transport net
 - * (for example introduction of toll roads or freeways,
 - * road pricing,
 - * major bridges across wide waters [potentially connecting two hitherto unconnected nets],
 - * and their management)
 are executive actions.
 - So are decisions on merging or splitting transport from or into several transport services.

Definition: Management & Organisation.

- *The composite term management and organisation*
 - applies in connection with *management* as outlined just above and
 - with *organisation* also outlined above.
- *The term then emphasises the relations between the organisation and management of an enterprise.*



The borderlines within management actions and across organisation “layouts” are fuzzy.

- Reorganising an enterprise
 - * from one characterised by a “deep” hierarchy of management layers (a hierarchy which may very well exemplify highly centralised both administrative and functional monitoring and control)
 - * into a matrix of two “shallow” hierarchies, one which addresses tactical and operational management and one which addresses executive and strategic management — with the former (the operations) being replicated across geographical areas while the latter applies “globally” —
 such reorganisations reflect executive actions (but are carried out by strategic and tactical management).

- **Strategic actions:** Adding or removing transport links, or major reorganisation of bus timetables are strategic actions. Splitting a(n own) contract into what is still to be operated and subcontracting other parts, for definite, to other bus operators are also strategic actions.
- **Tactical actions:** Insertion and cancellation of bus services are tactical actions. Subcontracting some parts of a timetable demanded service, for a short while, to other bus operators could be considered tactical actions.
- **Operational actions:** Commencing and thus, in general, allocating drivers to and sending these off on bus services are operational actions. So are announcing insertion of new (unscheduled) and cancellation of scheduled routes.

3.6. Human Behaviour

Definition: Human Behaviour.

- *By human behaviour we shall here understand*
 - *the way a human follows the enterprise rules and regulations*
 - *as well as interacts with a machine:*
 - * *dutifully honouring specified (machine dialogue or) protocols,*
 - * *or negligently so,*
 - * *or sloppily not quite so,*
 - * *or even criminally not so!*
- *Human behaviour is a facet of the domain.*
 - *We shall thus model human behaviour also in terms of it failing to react properly,*
 - *i.e., humans as non-deterministic agents!*

3.7. Towards Theories of Domain Facets

3.7.1. A Theory of Intrinsic

- For all intrinsic traffics, **itf**, and for all optical sensor technologies, **og**, the following must hold:
 - Let **stf** be the traffic sampled by the optical gates.
 - For all time points, **t**, in the sampled traffic,
 - those time points must also be in the intrinsic traffic,
 - and, for all trains, **tn**, in the intrinsic traffic at that time,
 - the train must be observed by the optical gates, and
 - the actual position of the train and the sampled position must somehow be checkable to be close, or identical to one another.

Since hubs change state with time, $n:N$, the net needs to be part of any model of traffic.

3.7.2. Theories of Support Technologies

3.7.2.1. An Example

- Traffic (**tf:TF**), intrinsically, is a total function over some time interval, from time (**t:T**) to continuously positioned (**p:P**) vehicles (**tn:TN**).
- Conventional optical sensors sample, at regular intervals, the intrinsic train traffic.
- The result is a sampled traffic (**stf:sTF**).
- Hence the collection of all optical sensors, for any given net, is a partial function from intrinsic (**itf**) to sampled train traffics (**stf**).
- We need to express quality criteria that any optical sensor technology should satisfy — relative to a necessary and sufficient description of a **closeness** predicate.

type

T, TN
 $P = HP \mid LP$
 $NetTraffic :: net:N \times trf:(V \rightsquigarrow P)$
 $iTF = T \rightarrow NetTraffic$
 $sTF = T \rightsquigarrow NetTraffic$
 $oG = iTF \rightsquigarrow sTF$

value

$[close] c: NetTraffic \times TN \times NetTraffic \rightsquigarrow \mathbf{Bool}$

axiom

$\forall itt:iTF, og:OG \cdot \mathbf{let} stt = og(itt) \mathbf{in}$
 $\forall t:T \cdot t \in \mathbf{dom} stt \cdot$
 $t \in \mathbf{DOM} itt \wedge \forall Tn:TN \cdot tn \in \mathbf{dom} trf(itt(t))$
 $\Rightarrow tn \in \mathbf{dom} trf(stt(t)) \wedge c(itt(t), tn, stt(t)) \mathbf{end}$

3.7.2.2. General

- The formal requirements can be narrated:
 - Let Θ_i and Θ_a designate the spaces of intrinsic and actual-world configurations (contexts and states).
 - For each intrinsic configuration model — that we know is support technology assisted —
 - there exists a support technology solution,
 - that is, a total function from all intrinsic configurations to corresponding actual configurations.
- If we are not convinced that there is such a function then there is little hope that we can trust this technology

type

$$\Theta_i, \Theta_a$$

$$ST = \Theta_i \rightarrow \Theta_a$$

axiom

$$\forall sts:ST\text{-set}, st:ST \cdot st \in sts \Rightarrow \forall \theta_i:\Theta_i, \exists \theta_a:\Theta_a \cdot st(\theta_i) = \theta_a$$

- A syntactic stimulus, **sy_sti**, denotes a function, **se_sti:STI**: $\Theta \rightarrow \Theta$, from any configuration to a next configuration
- A syntactic rule, **sy_rul:Rule**, has as its semantics, its meaning, **rul:RUL**,
 - a predicate over current and next configurations, $(\Theta \times \Theta) \rightarrow \mathbf{Bool}$,
 - where these next configurations have been caused, by the stimuli. These stimuli express:
 - If the predicate holds then the stimulus will result in a valid next configuration.

3.7.3. A Theory of Rules & Regulations

- There are, abstractly speaking, usually three kinds of languages involved wrt. (i.e., when expressing) rules and regulations (respectively when invoking actions that are subject to rules and regulations).
 - Two languages, **Rules** and **Reg**, exist for describing rules, respectively regulations; and
 - one, **Stimulus**, exists for describing the form of the [always current] domain action stimuli.

type

$$\text{Stimulus, Rule, } \Theta$$

$$STI = \Theta \rightarrow \Theta$$

$$RUL = (\Theta \times \Theta) \rightarrow \mathbf{Bool}$$

value

$$\text{meaning: Stimulus} \rightarrow STI$$

$$\text{meaning: Rule} \rightarrow RUL$$

$$\text{valid: Stimulus} \times \text{Rule} \rightarrow \Theta \rightarrow \mathbf{Bool}$$

$$\text{valid}(\text{sy_sti}, \text{sy_rul})(\theta) \equiv \text{meaning}(\text{sy_rul})(\theta, (\text{meaning}(\text{sy_sti}))(\theta))$$

$$\text{valid: Stimulus} \times RUL \rightarrow \Theta \rightarrow \mathbf{Bool}$$

$$\text{valid}(\text{sy_sti}, \text{se_rul})(\theta) \equiv \text{se_rul}(\theta, (\text{meaning}(\text{sy_sti}))(\theta))$$

- A syntactic regulation, **sy_reg:Reg** (related to a specific rule), stands for, i.e., has as its semantics, its meaning,
 - a semantic regulation, **se_reg:REG**,
 - which is a pair.
 - This pair consists of
 - * a predicate, **pre_reg:Pre_REG**, where $\text{Pre_REG} = (\Theta \times \Theta) \rightarrow \mathbf{Bool}$,
 - * and a domain configuration-changing function, **act_reg:Act_REG**, where $\text{Act_REG} = \Theta \rightarrow \Theta$,
 - * that is, both involving current and next domain configurations.

type

Reg

Rul_and_Reg = Rule \times RegREG = Pre_REG \times Act_REGPre_REG = $\Theta \times \Theta \rightarrow \mathbf{Bool}$ Act_REG = $\Theta \rightarrow \Theta$ **value**interpret: Reg \rightarrow REG

- The two kinds of functions express:
 - * If the predicate holds,
 - * then the action can be applied.
- The predicate is almost the inverse of the rules functions.
- The action function serves to undo the stimulus function.

- The idea is now the following:
 - Any action of the system, i.e., the application of any stimulus,
 - * may be an action in accordance with the rules,
 - * or it may not.
 - Rules therefore express whether stimuli are valid or not in the current configuration.
 - And regulations therefore express whether they should be applied, and, if so, with what effort.

- More specifically,
 - there is usually, in any current system configuration, given a set of pairs of rules and regulations.
 - Let $(\mathbf{sy_rul}, \mathbf{sy_reg})$ be any such pair.
 - Let $\mathbf{sy_sti}$ be any possible stimulus.
 - And let θ be the current configuration.
 - Let the stimulus, $\mathbf{sy_sti}$, applied in that configuration result in a next configuration, θ' , where $\theta' = (\mathbf{meaning}(\mathbf{sy_sti}))(\theta)$.
 - Let θ' ($= (\mathbf{meaning}(\mathbf{sy_sti}))(\theta)$) violate the rule, i.e., $\sim\mathbf{valid}(\mathbf{sy_sti}, \mathbf{sy_rul})(\theta)$
 - then if predicate part, $\mathbf{pre_reg}$, of the meaning of the regulation, $\mathbf{sy_reg}$, holds in that violating next configuration, $\mathbf{pre_reg}(\theta, \theta'$
 - then the action part, $\mathbf{act_reg}$, of the meaning of the regulation, $\mathbf{sy_reg}$, must be applied, $\mathbf{act_reg}(\theta')$, to remedy the situation.

- It may be that the regulation predicate fails to detect applicability of regulations actions.
- That is, the interpretation of a rule differs, in that respect, from the interpretation of a regulation.
- Such is life in the domain, i.e., in actual reality

axiom

$$\begin{aligned} & \forall (\mathbf{sy_rul}, \mathbf{sy_reg}): \mathbf{Rul_and_Regs} \cdot \\ & \quad \mathbf{let} \ \mathbf{se_rul} = \mathbf{meaning}(\mathbf{sy_rul}), \\ & \quad (\mathbf{pre_reg}, \mathbf{act_reg}) = \mathbf{meaning}(\mathbf{sy_reg}) \ \mathbf{in} \\ & \forall \ \mathbf{sy_sti}: \mathbf{Stimulus}, \ \theta: \Theta \cdot \\ & \quad \sim\mathbf{valid}(\mathbf{sy_sti}, \mathbf{se_rul})(\theta) \\ & \quad \Rightarrow \mathbf{let} \ \theta' = (\mathbf{meaning}(\mathbf{sy_sti}))(\theta) \ \mathbf{in} \\ & \quad \quad \mathbf{pre_reg}(\theta, \theta') \\ & \quad \quad \Rightarrow \exists \ n\theta: \Theta \cdot \mathbf{act_reg}(\theta') = n\theta \wedge \mathbf{se_rul}(\theta, n\theta) \\ & \mathbf{end \ end} \end{aligned}$$

3.7.4. A Theory of Management & Organisation

3.7.5. A Theory of Human Behaviour

- Commensurate with the above, humans interpret rules and regulations differently,
- and not always “consistently” — in the sense of repeatedly applying the same interpretations.
- Our final specification pattern is therefore:

- The above is, necessarily, sketchy:
 - There is a possibly infinite variety of ways of interpreting some rules.
 - A human, in carrying out an action, interprets applicable rules and chooses one which that person believes suits some (professional, sloppy, delinquent or criminal) intent.
 - “Suits” means that it satisfies the intent,
 - * i.e., yields **true** on the pre/post-configuration pair,
 - * when the action is performed —
 - * whether as intended by the ones who issued the rules and regulations or not.
 - We do not cover the case of whether an appropriate regulation is applied or not

type

Action = $\Theta \xrightarrow{\sim} \Theta$ -infset

value

hum_int: Rule $\rightarrow \Theta \rightarrow$ RUL-infset

action: Stimulus $\rightarrow \Theta \rightarrow \Theta$

hum_beha: Stimulus \times Rules \rightarrow Action $\rightarrow \Theta \xrightarrow{\sim} \Theta$ -infset

hum_beha(sy_sti,sy_rul)(α)(θ) as θ set

post

θ set = $\alpha(\theta) \wedge$ action(sy_sti)(θ) \in θ set

$\wedge \forall \theta': \Theta \cdot \theta' \in \theta$ set \Rightarrow

\exists se_rul: RUL-se_rul \in hum_int(sy_rul)(θ) \Rightarrow se_rul(θ, θ')

End of Lecture 4: DOMAINS: Scripts – Human Behaviour

B Slide Table-of-Contents

Contents

Lect. # 1: COVER & INTRODUCTION	0
1 Introduction	5
1.1 The Problem	5
1.2 The Triptych Approach	6
Lect. # 2: ONTOLOGY	8
2 An Ontology of Specification Entities	9
2.1 Simple Entities	11
2.1.1 Net, Hubs and Links	13
2.1.2 Unique Hub and Link Identifiers	14
2.1.3 Observability of Hub and Link Identifiers	15
2.1.4 A Theorem	17
2.1.4.1 Links implies Hubs	17
2.1.5 Hub and Link Attributes	18
2.1.6 Hub and Link Generators	19
2.2 States	22
2.3 Actions	22
2.3.1 Insert Hubs	23
2.3.2 Remove Hubs	25
2.3.3 Insert Links	27
2.3.4 Remove Links	31
2.3.5 Two Theorems	34
2.3.5.1 Idempotency	34
2.3.5.2 Reachability	35
2.4 Events	37
2.5 Behaviours	41
2.5.1 Behaviour Prescriptions	42
2.5.1.1 Construction Plans	42

Lect. # 4: DOMAINS: Scripts – Human Behaviour	99
3.4 Scripts	100
3.4.1 Routes as Scripts	101
3.4.1.1 Paths	101
3.4.1.2 Routes	104
3.4.2 Bus Timetables as Scripts	107
3.4.2.1 Buses	107
3.4.2.2 Bus Stops	107
3.4.2.3 Bus Routes	108
3.4.2.4 Bus Schedule	110
3.4.2.5 Timetable	112
3.4.3 Route and Bus Timetable Denotations	115
3.4.4 Licenses and Contracts	117
3.4.4.1 Contracts	123
3.4.4.2 Contractual Actions	128
3.4.4.3 Wellformedness of Contractual Actions	131
3.5 Management and Organisation	137
3.5.1 Transport System Examples	141
3.6 Human Behaviour	144
3.7 Towards Theories of Domain Facets	146
3.7.1 A Theory of Intrinsic	147
3.7.2 Theories of Support Technologies	148
3.7.2.1 An Example	148
3.7.2.2 General	151
3.7.3 A Theory of Rules & Regulations	152
3.7.4 A Theory of Management & Organisation	162
3.7.5 A Theory of Human Behaviour	163
Lect. # 5: REQUIREMENTS – up to and incl. Determination	165
4 An Ontology of Requirements Constructions	166
4.1 Business Process Re-engineering	169
4.1.1 The Kinds of Requirements	172
4.1.2 Goals Versus Requirements	173
4.1.2.1 Goals of a Toll Road System	175

2.5.1.2 Wellformedness of Construction Plans	43
2.5.2 Augmented Construction Plans	47
2.5.3 Sequential Construction Behaviours	50
Lect. # 3: DOMAINS: Intrinsic – Rules & Regulations	52
3 An Ontology of Domain Facets	53
3.0.1 Definitions	53
3.0.2 What Can Be Observed	57
3.0.3 Business Processes	58
3.0.3.1 A Characterisation	58
3.0.3.2 An Example	58
3.1 Intrinsic	59
3.1.1 Net Topology Descriptors	60
3.1.2 Link States and Link State Spaces	63
3.1.3 Hub States and Hub State Spaces	67
3.1.4 State and State Space Wellformedness	69
3.1.5 Concrete Types for Simple Entities	70
3.1.6 Example Hub Crossings	73
3.1.7 Actions Continued	75
3.2 Support Technologies	77
3.2.1 Traffic Signals	78
3.2.2 Traffic “Control”	82
3.3 Rules and Regulations	84
3.3.1 Vehicles	87
3.3.2 Traffic	89
3.3.2.1 Wellformedness of Traffic	89
3.3.2.1.1 Static Wellformedness	90
3.3.2.1.2 Dynamic Wellformedness	92
3.3.3 Traffic Rules (I of II)	97
3.3.4 Another Traffic Regulator	98
3.3.5 Traffic Rules (II of II)	99

4.1.2.2 Goals of Toll Road System Software	176
4.1.2.3 Arguing Goal-satisfaction of a Toll Road System	177
4.1.2.4 Arguing Goal-satisfaction of Toll Road System Software	178
4.1.3 Re-engineered Nets	180
4.2 Domain Requirements	191
4.2.1 Projection	193
4.2.1.1 Example	195
4.2.2 Instantiation	196
4.2.2.1 Example	197
4.2.2.2 Abstraction: From Concrete Toll Road Nets to Abstract Nets	202
4.2.2.3 Theorem	203
4.2.3 Determination	204
4.2.3.1 Example	205
Lect. # 6: REQUIREMENTS – from Extension “out”	208
4.2.4 Extension	209
4.2.4.1 Intuition	212
4.2.4.2 Descriptions	214
4.2.4.2.1 A RAISE/CSP Model	214
4.2.4.2.1 Toll Booth Plazas	214
4.2.4.2.1 Cars	216
4.2.4.2.1 Entry Booths	217
4.2.4.2.1 Gates	219
4.2.4.2.1 The Entry Plaza System	220
4.2.4.2.2 A Duration Calculus Model	225
4.2.4.2.3 A Timed Automata Model	229
4.2.5 Fitting	233
4.2.5.1 Examples	234
4.3 Interface Requirements	235
4.3.1 But First: On Shared Phenomena and Concepts	237
4.3.2 Shared Simple Entities	238
4.3.2.1 Example	239
4.3.3 Shared Actions	240
4.3.3.1 Example	241
4.3.4 Shared Events	242

4.3.4.1 Examples	243
4.3.5 Shared Behaviours	244
4.3.5.1 Example	245
4.4 Machine Requirements	246
4.4.1 An Enumeration of Classes of Machine Requirements	247

Lect. # 11: CLOSING **247**

5 Conclusion	248
5.1 What Have We Omitted	248
5.2 Domain Descriptions Are Not Normative	249
5.3 "Requirements Always Change"	250
5.4 What Can Be Described and Prescribed	252
5.5 What Have We Achieved – and What Not	254
5.6 Relation to Other Work	255
5.7 "Ideal" Versus Real Developments	258
5.8 Description Languages	260
5.9 Entailments	262
5.10 Domain Versus Ontology Engineering	263

6 Bibliographical Notes	264
6.1 Description Languages	264

Lect. # 7: RSL: Types **265**

1 An RSL Primer	266
1.1 Types	266
1.1.1 Type Expressions	266
1.1.1.1 Atomic Types	266
Example 1: Basic Net Attributes	268
1.1.1.2 Composite Types	270
Example 2: Composite Net Type Expressions	271
1.1.2 Type Definitions	273
1.1.2.1 Concrete Types	273
Example 3: Composite Net Types	274
Example 4: Net Record Types: Insert Links	280

1.1.2.2 Subtypes	282
Example 5: Net Subtypes	283
1.1.2.3 Sorts — Abstract Types	289
Example 6: Net Sorts	290

Lect. # 8: RSL: Values & Operations **290**

1.2 Concrete RSL Types: Values and Operations	291
1.2.1 Arithmetic	291
1.2.2 Set Expressions	292
1.2.2.1 Set Enumerations	292
Example 7: Set Expressions over Nets	293
1.2.2.2 Set Comprehension	297
Example 8: Set Comprehensions	298
1.2.3 Cartesian Expressions	299
1.2.3.1 Cartesian Enumerations	299
Example 9: Cartesian Net Types	300
1.2.4 List Expressions	303
1.2.4.1 List Enumerations	303
1.2.4.2 List Comprehension	304
Example 10: Routes in Nets	305
1.2.5 Map Expressions	310
1.2.5.1 Map Enumerations	310
1.2.5.2 Map Comprehension	311
Example 11: Concrete Net Type Construction	312
1.2.6 Set Operations	315
1.2.6.1 Set Operator Signatures	315
1.2.6.2 Set Examples	316
1.2.7 Cartesian Operations	317
1.2.8 List Operations	318
1.2.8.1 List Operator Signatures	318
1.2.8.2 List Operation Examples	319
1.2.9 Map Operations	320
1.2.9.1 Map Operator Signatures and Map Operation Examples	320

Lect. # 9: RSL: Logic, λ-Calculus, Fctl. Specs.	321
1.3 The RSL Predicate Calculus	322
1.3.1 Propositional Expressions	322
1.3.2 Simple Predicate Expressions	323
1.3.3 Quantified Expressions	324
Example 12: Predicates Over Net Quantities	325
1.4 λ -Calculus + Functions	328
1.4.1 The λ -Calculus Syntax	328
1.4.2 Free and Bound Variables	329
1.4.3 Substitution	330
1.4.4 α -Renaming and β -Reduction	331
Example 13: Network Traffic	332
1.4.5 Function Signatures	339
Example 14: Hub and Link Observers	340
1.4.6 Function Definitions	342
Example 15: Axioms over Hubs, Links and Their Observers	345
1.5 Other Applicative Expressions	346
1.5.1 Simple Iet Expressions	346
1.5.2 Recursive Iet Expressions	347
1.5.3 Non-deterministic Iet Clause	348
1.5.4 Pattern and "Wild Card" Iet Expressions	349
1.5.5 Conditionals	350
Example 16: Choice Pattern Case Expressions: Insert Links	351
1.5.6 Operator/Operand Expressions	361

Lect. # 10: RSL: Imperative & Process Specs. **361**

1.6 Imperative Constructs	362
1.6.1 Statements and State Changes	362
1.6.2 Variables and Assignment	363
1.6.3 Statement Sequences and skip	363
1.6.4 Imperative Conditionals	363
1.6.5 Iterative Conditionals	364
1.6.6 Iterative Sequencing	364
1.7 Process Constructs	365
1.7.1 Process Channels	365

Example 17: Modelling Connected Links and Hubs	366
1.7.2 Process Definitions	370
Example 18: Communicating Hubs, Links and Vehicles	372
1.7.3 Process Composition	374
Example 19: Modelling Transport Nets	375
1.7.4 Input/Output Events	378
Example 20: Modelling Vehicle Movements	379
1.8 Simple RSL Specifications	385
Example 21: A Neat Little "System"	389

B Slide Table-of-Contents **398**